IN THE CLAIMS:

Claim 1 (Currently Amended): A method for managing ~~a~~an original user interface for ~~in a multithreaded computing environment, the user interface comprising a plurality of user interface elements wherein a first user interface element in the plurality of user interface elements corresponds to~~ a~~a~~n ~~first~~ application, ~~wherein furthermore the first application having a first thread having control over the first user interface element,~~ the method comprising ~~the steps of~~:

A3

signaling a hung state for the ~~first~~ application; ~~if there is a delay graeater than a prepdetermined threshold in handling events queued in an event queue for handling by the first application;~~

creating a ghost user interface ~~element, with a ghost thread,~~in response to the hung state ~~responsively to a hung signal indicating the first application is in the hung state~~ that appears in place of the original user interface~~wherein the ghost user interface element replaces the first user interface element in the user interface~~;

detecting activity ~~placing a high priority special event~~ in a~~n~~ ~~the~~ event queue of ~~for handling by~~ the ~~first~~ application ~~wherein handling of the special event generates a activity detect message~~; and

~~detecting the wakeup message and responsively to the activity detect message~~ replacing the ghost user interface ~~element by~~ with the original ~~first~~ user interface ~~element~~ in response to the detection of activity.

Claim 2 (Currently Amended): The method of claim 1 wherein ~~the step of signaling includes~~ signaling a hung state occurs if the ~~first~~ application does not handle an event in ~~an~~ the event queue for at least a predetermined duration.

Claim 3 (Currently Amended): The method of claim 1 ~~wherein the step of detecting the activity detect message~~ further ~~comprises~~comprising releasing~~, responsively to the activity detect message,~~ resources used by the ghost user interface in response to detecting activity in the event queue~~element~~.

Claim 4 (Currently Amended): The method of claim 1 wherein ~~the step of~~ creating the ghost user interface ~~element~~ includes:

~~creating the a ghost thread, which, in turn, creates the ghost user interface element~~

creating a ghost thread for the ghost user interface.

Claim 5 (Original): The method of claim 4 wherein ~~the step of~~ creating the ghost thread includes determining if a ghost thread exists and creating a ghost thread if the ghost thread does not exist.

A3

Claim 6 (Currently Amended): The method of claim ~~1~~ 4 wherein ~~the step of~~ creating the ghost user interface ~~element~~ includes creating the ghost user interface ~~element~~ in the area occupied by the original ~~the first~~ user interface ~~element~~.

Claim 7 (Currently Amended): The method of claim 6~~, the method~~ further comprising ~~having the step of~~ directing input events corresponding to the area covered by the ghost user interface ~~element~~, which includes at least some of the area covered by the original ~~first~~ user interface ~~element~~, to the ghost thread.

Claim 8 (Currently Amended): The method of claim 7~~, the method~~ further comprising ~~having the step of~~ having the ghost thread handle at least one event corresponding to the area covered by the ghost user interface.

Claim 9 (Currently Amended): The method of claim 7 further comprising ~~having the step of~~ having the ghost thread cache at least one event corresponding to the area covered by the ghost user interface to the ghost thread for handling by the ~~first~~ application.

Claim 10 (Currently Amended): The method of claim 7~~, the method~~ further comprising ~~having the step of~~ having the ghost thread handle at least one event

corresponding to a minimization operation in the area covered by the ghost user interface ~~element~~.

Claim 11 (Currently Amended): The method of claim 7~~, the method~~ further comprising ~~having the step of~~ having the ghost thread handle at least one event corresponding to a resizing operation in the area covered by the ghost user interface ~~element~~.

Claim 12 (Currently Amended): The method of claim 7~~, the method~~ further comprising ~~having the step of~~ having the ghost thread handle at least one event corresponding to a close operation in the area covered by the ghost user interface ~~element~~.

A3

Claim 13 (Currently Amended): The method of claim 7~~, the method~~ further comprising ~~having the step of~~ having the ghost thread handle at least one event corresponding to a move operation in the area covered by the ghost user interface ~~element~~.

Claim 14 (Currently Amended): The method of claim 7~~, the method~~ further comprising ~~having the step of~~ having the ghost thread cache at least one event corresponding to a keyboard input corresponding to the ghost user interface ~~element~~.

Claim 15 (Currently Amended): The method of claim 7~~, the method~~ further comprising ~~having the step of~~ having the ghost thread handle at least one event corresponding to a keyboard input corresponding to the ghost user interface ~~element~~.

Claim 16 (Currently Amended): A method for replacing a ~~first~~ user interface element created by a ~~first~~ application by a ~~first~~ substitute user interface element created by a scheduled code path in a multithreaded computing environment, the method comprising ~~the steps of~~:

detecting a ~~first~~ flip-window signal while the ~~first~~ user interface element is being displayed;

~~creating, responsively~~ in response to the ~~first~~ flip-window signal, creating the ~~first~~ substitute user interface element in the context of the scheduled code path;

superimposing the ~~first~~ substitute user interface element over a first-user-interface-element-occupied real estate; and

caching input corresponding to the ~~first~~ substitute user interface element wherein the cached input is subsequently handled by the ~~first~~ application.

A3

Claim 17 (Original): The method of claim 16 wherein the step of creating includes creating the scheduled code path if the scheduled code path does not exist.

Claim 18 (Currently Amended): The method of claim 16~~, the method~~ further comprising ~~having the step of~~ replacing a second user interface element created by a second application by a second substitute user interface element created by the scheduled code path responsively to a second flip-window signal.

Claim 19 (Currently Amended): The method of claim 16~~, the method~~ further comprising ~~having the step of~~ painting over the ~~first~~ user interface element responsively to the ~~first~~ flip-window signal.

Claim 20 (Currently Amended): The method of claim 16 wherein ~~the step of~~ caching input ~~forwarding~~ includes input from ~~comprising~~ a plurality of events and/or a plurality of messages.

Claim 21 (Currently Amended): The method of claim 16 ~~17, the method~~ further comprising ~~having the step of~~ caching input corresponding to the application for forwarding to the application in response to the ~~a first~~ flop-window signal.

Claim 22 (Currently Amended): The method of claim 18~~, the method~~ further comprising ~~having the steps of~~ selecting and discarding at least some of the cached input.

Claim 23 (Currently Amended): The method of claim 16 ~~19~~ wherein ~~the step of~~ caching input ~~forwarding~~ includes forwarding the input to the scheduled code path.

Claim 24 (Currently Amended): The method of claim 20~~, the method~~ further comprising ~~having the step of~~ handling at least some of the input by the scheduled code path.

AB

Claim 25 (Currently Amended): The method of claim 24~~, the method~~ further comprising ~~having the step of~~ the scheduled code path handling at least one event in the input by terminating the ~~first~~ application.

Claim 26 (Currently Amended): The method of claim 16~~, the method~~ further comprising ~~having the step of~~ placing a special message or event ~~message/event~~ in a queue for the ~~first~~ application to generate a flop-window signal.

Claim 27 (Original): The method of claim 16 wherein the scheduled code path is a thread.

Claim 28 (Currently Amended): A method ~~of using a designated scheduled code path for providing~~ for ~~substitute~~ substituting user interfaces for original ~~replacing~~ user interfaces ~~corresponding to a plurality of scheduled code paths~~, the method comprising ~~the steps of~~:

generating ~~a~~ first and second flip-window signals corresponding to ~~a~~ first and second threads, respectively ~~scheduled code path~~;

~~generating a second flip-window signal corresponding to a second scheduled code path;~~

~~replacing, responsively~~ in response to the first flip-window signal, replacing a first window controlled by the first thread ~~scheduled code path~~ with a first substitute window controlled by a thread designated to shadow an appearance of the first window ~~the designated scheduled code path~~; and

~~replacing, responsively~~ in response to the second flip-window signal, replacing a second window controlled by the second thread ~~scheduled code path~~ with a second substitute window controlled by the designated thread~~scheduled code path~~.

Claim 29 (Currently Amended): The method of claim 28~~, the method~~ further comprising ~~having the steps of~~:

A3

generating a first flop-window signal; and

~~replacing, responsively~~ in response to the first flop-window signal, replacing a first substitute window controlled by the designated thread ~~scheduled code path~~ with the first window controlled by the first thread ~~scheduled path~~.

Claim 30 (Currently Amended): A multithreaded computing system for executing an ~~a first~~ application having a ~~first~~ user interface, wherein if the ~~first~~ application is non-responsive to user input, the ~~first~~ user interface is replaced by a ~~first~~ ghost user interface, the system comprising:

a non-responsive-application detecting code for detecting a non-responsive application;

a ghost thread for creating the ~~first~~ ghost user interface to replace the ~~first~~ user interface in response ~~responsively~~ to detection of a non-responsive application;

a high priority special entry in a queue for the ~~first~~ application for detecting if the ~~first~~ application is responsive; and

a plurality of computer executable instructions for destroying the ~~first~~ ghost user interface in response ~~responsively~~ to handling of the high priority special entry by the ~~first~~ application.

Claim 31 (Currently Amended): The system of claim 30 further comprising ~~having~~ a cache of events and messages for handling by the ~~first~~ application wherein the events and messages are directed at the ghost thread.

Claim 32 (Currently Amended): The system of claim 30 further comprising ~~having~~ a second non-responsive application with a second user interface and a second ghost user interface created by the ghost thread.

Claim 33 (Currently Amended): The system of claim 30 further comprising ~~having~~ a responsive-application detecting code for detecting when a non-responsive application becomes responsive.

A3

Claim 34 (Currently Amended): The system of claim 33 further comprising ~~having~~ a responsive-application user interface restoring code for replacing the first ghost user interface with the first user interface in response ~~responsively~~ to detecting that the first application has become responsive.

Claim 35 (Currently Amended): A ~~computer-readable~~ computer-readable media having computer executable instructions for carrying out the steps of a method for managing an original ~~a~~ user interface for ~~in a multithreaded computing environment, the user interface comprising a plurality of user interface elements wherein a first user interface element in the plurality of user interface elements corresponds to a first~~ an application, ~~wherein furthermore the first application having a first thread having control over the first user interface element,~~ the method comprising ~~the steps of~~:

signaling a hung state for the ~~first~~ application; ~~(if there is a delay greater than a predetermined threshold in handling events queued in an event queue for handling by the first application)~~;

creating a ghost user interface ~~element, with a ghost thread,~~in response to the hung state ~~responsively to a hung signal indicating the first application is in the hung~~

~~state~~ that appears in place of the original user interface~~wherein the ghost user interface element replaces the first user interface element in the user interface~~;

detecting activity ~~placing a high priority special event~~ in an ~~the~~ event queue of ~~for~~ ~~handling by~~ the ~~first~~ application ~~wherein handling of the special event generates a activity detect message~~; and

A3

~~detecting the wakeup message and responsively to the activity detect message~~ replacing the ghost user interface ~~element by~~ with the ~~first~~ user interface ~~element~~ in response to the detection of activity.